

DOCUMENT RANKING BASED ON SEMANTIC
DISTANCE BETWEEN TERMS IN A DOCUMENT

BACKGROUND OF THE INVENTION

A. Field of the Invention

[0001] The present invention relates generally to information retrieval and, more particularly, to automated techniques for ranking documents in response to a search query.

B. Description of Related Art

[0002] The World Wide Web ("web") contains a vast amount of information.

Search engines assist users in locating desired portions of this information by cataloging web pages. Typically, in response to a user's request, the search engine returns references to documents relevant to the request.

[0003] Search engines may base their determination of the user's interest on search terms (called a search query) entered by the user. The goal of the search engine is to identify links to high quality relevant results based on the search query. Typically, the search engine accomplishes this by matching the terms in the search query to a corpus of pre-stored web documents. Web documents that contain the user's search terms are considered "hits" and are returned to the user.

[0004] It may be desirable to rank the hits returned by the search engine based on some measure of the quality and relevancy of the hits. A basic technique for sorting the search engine hits relies on the degree with which the search query matches the hits. For example, documents that contain every term of the search query or that contain multiple occurrences of the terms in the search query may

be deemed more relevant than other documents and therefore may be more highly ranked by the search engine. Other factors, such as the closeness of terms (also referred to as distance between the terms) in the document may also be considered. Closeness of terms in this context may be measured simply by counting the number of words in the document occurring between the search terms. In documents such as web pages, however, which may contain complex formatting information, "closeness" of terms in the underlying HTML file may not correlate with the "closeness" of the terms when the document is visually displayed. Accordingly, the performance of search engines that rank documents based on the closeness of the search terms in the underlying documents can suffer.

[0005] For search engines, returning relevant and high quality documents in response to a search query is of paramount importance. Accordingly, it would be desirable to improve search engine ranking techniques that consider closeness of terms in a search query in the underlying document.

SUMMARY OF THE INVENTION

[0006] One aspect of the invention includes a method of calculating a distance value between terms in a document. The method may include locating implicitly defined semantic structures in the document and calculating the distance values in the document based on the implicitly defined semantic structures.

[0007] Another aspect consistent with the invention is a method for ranking documents relative to a search query. The method may include determining semantically based distance values between terms that occur in the search query

and that are present in the documents, and ranking the documents for relevancy to the search query based on the determined distance values.

[0008] Yet another aspect of the invention is a device that includes a memory and a processor coupled to the memory. The processor may determine semantically based distance values between terms that occur in a search query and that are present in documents. The processor may also rank the documents for relevancy to the search query based on the determined distance values.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate an embodiment of the invention and, together with the description, explain the invention. In the drawings,

[0010] Fig. 1 is a diagram of an exemplary overview illustrating the concept of semantic distance relative to visual distance;

[0011] Fig. 2 is an exemplary diagram of a network;

[0012] Fig. 3 is an exemplary diagram of a client or server;

[0013] Fig. 4 is a block diagram illustrating an implementation of the ranking component shown in Fig. 2;

[0014] Fig. 5A illustrates the HTML code for a list;

[0015] Fig. 5B illustrates an exemplary rendered version of the HTML code shown in Fig. 5A;

[0016] Fig. 6 is a flow chart illustrating an implementation for detecting implicit lists in a document;

[0017] Fig. 7 is a diagram illustrating an exemplary tree structure;

[0018] Fig. 8 is a flow chart illustrating operations according to an exemplary implementation for determining ranking scores for documents;

[0019] Fig. 9 illustrates concepts that may be used to augment a word count between terms when generating a distance metric; and

[0020] Fig. 10 is a diagram illustrating an exemplary implementation of a ranking component implemented in the context of an Internet search engine.

DETAILED DESCRIPTION

[0021] The following detailed description of the invention refers to the accompanying drawings. The detailed description does not limit the invention.

OVERVIEW

[0022] As described herein, a relevance component uses a measured “distance” between pairs of search query terms when determining relevance of documents to the search query. The distance may be measured based on how close together terms are semantically rather than visually. In other words, even if terms are visually far apart in the rendered document, the measured distance may still indicate that the words are relatively close if the semantic structure of the document indicates that the words are semantically close.

[0023] Fig. 1 is a diagram of an exemplary overview illustrating the concept of semantic distance relative to visual distance. Fig. 1 illustrates an exemplary document, such as a web page, including a list of facts about the planet Saturn. Semantically, each row of the list is probably equally relevant to every other row and to the title of the list, “Saturn Facts,” despite the fact that the various items are spaced at different visual distances from each other and from the title of the

list. Thus, from a semantic standpoint, the item "Mass is 95 times that of Earth" should be considered to be equally distant from the title of the list ("Saturn Facts") as the item "One Orbit of Sun is 10,759.2 Days."

EXEMPLARY NETWORK OVERVIEW

[0024] Fig. 2 is an exemplary diagram of a network 200 in which concepts consistent with the principles of the invention may be implemented. Network 200 may include multiple clients 210 connected to a server 220 via a network 240. Network 240 may include a local area network (LAN), a wide area network (WAN), a telephone network, such as the Public Switched Telephone Network (PSTN), an intranet, the Internet, or a combination of networks. Two clients 210 and one server 220 have been illustrated as connected to network 240 for simplicity. In practice, there may be more clients and/or servers. Also, in some instances, a client may perform the functions of a server and a server may perform the functions of a client.

[0025] Clients 210 may include devices such as a wireless telephone, a personal computer, a personal digital assistant (PDA), a lap top, or another type of computation or communication device, a thread or process running on one of these devices, and/or an object executable by one of these devices. Server 220 may include server devices that process, search, and/or maintain documents in a manner consistent with the principles of the invention. Clients 210 and server 220 may connect to network 240 via wired, wireless, or optical connections.

[0026] In one implementation, server 220 may implement a search engine (not shown) usable by clients 210. In general, in response to a client request, the search engine returns documents, such as web pages, to the client. These web

pages may be sorted and displayed in an order influenced by ranking scores calculated based on a distance between terms in the web page. In the implementation shown, ranking component 225 may generate the ranking scores used to sort the web pages.

[0027] A document, as the term is used herein, is to be broadly interpreted to include any machine-readable and machine-storable work product. A document may be an e-mail, a blog, a file, a combination of files, one or more files with embedded links to other files, a news group posting, etc. In the context of the Internet, a common document is a web page. Web pages often include content and may include embedded information (such as meta information, hyperlinks, etc.) and/or embedded instructions (such as Javascript, etc.).

EXEMPLARY CLIENT/SERVER ARCHITECTURE

[0028] Fig. 3 is an exemplary diagram of a client 210 or server 220. Client/server 210/220 may include a bus 310, a processor 320, a main memory 330, a read only memory (ROM) 340, a storage device 350, input devices 360, output devices 370, and a communication interface 380. Bus 310 may include conductors that permit communication among the components of client/server 210/220.

[0029] Processor 320 may include conventional processors or microprocessors that interpret and execute instructions. Main memory 330 may include a random access memory (RAM) or another type of dynamic storage device that stores information and instructions for execution by processor 320. ROM 340 may include a conventional ROM device or another type of static storage device that stores static information and instructions for use by processor 320. Storage

device 350 may include a magnetic and/or optical recording medium and its corresponding drive.

[0030] Input devices 360 may include one or more conventional mechanisms that permit a user to input information to client/server 210/220, such as a keyboard, a mouse, a pen, voice recognition and/or biometric mechanisms, etc. Output devices 370 may include one or more conventional mechanisms that output information to the user, including a display, a printer, a speaker, etc. Communication interface 380 may include any transceiver-like mechanism that enables client/server 210/220 to communicate with other devices and/or systems. For example, communication interface 380 may include mechanisms for communicating with another device or system via a network, such as network 240.

[0031] As will be described in detail below, server 220, consistent with the principles of the invention, may implement ranking component 225. Ranking component 225 may be stored in a computer-readable medium, such as memory 330. A computer-readable medium may be defined as one or more physical or logical memory devices and/or carrier waves.

[0032] The software instructions defining ranking component 225 may be read into memory 330 from another computer-readable medium, such as data storage device 350, or from another device via communication interface 380. The software instructions contained in memory 330 may cause processor 320 to perform processes that will be described later. Alternatively, hardwired circuitry or other logic may be used in place of, or in combination with, software instructions to implement processes consistent with the present invention. Thus,

implementations consistent with the principles of the invention are not limited to any specific combination of hardware circuitry and software.

RANKING COMPONENT 225

[0033] Fig. 4 is a block diagram illustrating an implementation of ranking component 225 in additional detail. Ranking component 225 may include a page analyzer component 410, a distance component 420, and a relevance component 430. Page analyzer component 410 may generally analyze documents, such as HTML-based (Hyper-Text Markup Language) web pages, to detect lists, including implicit lists, within the documents. Distance component 420 may implement a distance function that calculates distance metrics between pairs of terms in the document. The distance metrics may be semantically based distances, such that, for example, terms that are in different items of a list, terms that are structurally separated, or terms that are separated by many words are considered far apart. The distances determined by distance component 420 may be used by a relevance component 430 in determining the relevance of the document to the search query.

[0034] Structures other than lists may be detected by page analyzer component 410. For example, titles and headings may also be detected.

[0035] As previously mentioned, page analyzer component 410 may analyze documents to detect lists. When the documents are HTML documents, the lists can be determined based on HTML tags. Some lists may be explicitly defined by the HTML tags. For example, the HTML "" and "" tag are used to delimit the start and stop of an explicitly defined unordered list. An unordered list may be defined as a collection of items (typically related somehow) that are in no

particular order. The “``” tag is another HTML tag that explicitly defines a list.

More specifically, “``” and “``” delimit the start and stop of an ordered list.

[0036] HTML lists, in addition to being defined explicitly with the `` and `` tags, may be defined implicitly. Web authors frequently use HTML tags other than `` and `` to create lists. To create different layouts, they may, for example, use nested tables (using the “`<table>`” and “`<tr>`” tags) or other HTML tags, such as the “`<div>`” tags, “`
`” tags, and “`<p>`” tags to create lists. Other tags, in addition to these listed tags may be used to define an implicit list. Unlike `` and ``, however, these tags not formally defined for the creation of lists.

Rather, they are used by web authors in an ad-hoc manner to create lists.

[0037] Figs. 5A and 5B are diagrams illustrating an exemplary implicit list defined using the `
` (line break) tag and the `` (bold font) tag. Fig. 5A illustrates the HTML code 520 of the list and Fig. 5B illustrates an exemplary rendered version 500 of the list. As shown, rendered version 500 of the list includes a number of items 510-514. Each item, as shown in HTML code 520, is defined using the `` tag to cause the first word in the list to be a bold font and ends with a `
` tag to create a line break.

[0038] Page analyzer component 410 may decompose a document, such as the HTML code 520, to analyze the structure of the document. In one implementation, page analyzer component 410 decomposes the document into a tree structure using the HTML markup (e.g., HTML tags) information. Implicit lists may be determined based on the repetition of the HTML tags in the tree structure.

OPERATION OF PAGE ANALYZER COMPONENT 410

[0039] Fig. 6 is a flow chart illustrating exemplary operations of page analyzer component 410 for detecting semantic structures in a document, such as HTML code 520, in additional detail. The document may be initially parsed into a tree structure (act 601).

[0040] Fig. 7 is a diagram illustrating an exemplary tree structure 700 created from HTML code 520. Tree structure 700 may display the hierarchical tag markup structure of HTML code 520. As shown, tree structure 700 includes DIV tag 710, which is an HTML tag used to divide a document into sections and specify alignments for the sections. In this example, DIV tag 710 specifies center alignment (see Fig. 5A). Below DIV tag 710 are a number of tags, including the font tag 712, which includes the beginning font tag `` and the ending font tag ``. Font tag 712 specifies the font size for the text “Saturn Facts.” Tree structure 700 includes a first line break tag 713 (`
`). Tree structure 700 next includes bold tag 714 (``). Bold tag 714 and line break tag 713 are repeated for each item in HTML code 520 to thus create an implicit list. Visually, the combination of the bold font and the line breaks make clear to the reader that this text is a list of items.

[0041] The repetition of commands in tree structure 700 may be analyzed to detect implicit lists (act 602). In one implementation, the analysis may be performed by looking for sets of two or more text formatting commands that are repeated. The potential text formatting commands may include the HTML tags `<div>`, `
`, and `<p>`, or that include table related HTML tags. One of ordinary skill in the art will recognize that other text formatting commands, in addition

these commands, could be used to create implicit lists and could be included in the potential text formatting commands. Additionally, other techniques, or variations on the technique of looking for sets of two or more text formatting commands, may be used to detect implicit lists. Additionally, special rules can be used to handle the first or last element of a list. For example, in tree structure 700, the first
 tag 713 may be missing. The special rules may be designed to handle these situations and still identify the list.

[0042] In the example shown in Fig. 7, each set of
 tags 713 and
 tags 714 may be recognized as a set of repeating tags in act 602 that defines an item of an implicit list. Tree structure 700 of HTML code 520, or the document itself, may be annotated to indicate the presence of this implicit list (act 603). Other lists, such as explicit lists specified with or tags may also be annotated.

[0043] In addition to lists, page analyzer component 410 may use tree structure 700 to detect headings and titles (act 602). Headings and titles can be detected in tree structure 700 as text associated with nodes that are above other nodes in the tree structure. In other words, text below a header/title node in the tree structure can be considered to belong to that heading/title.

[0044] As discussed above, page analyzer component 410 locates visual structures that are semantically meaningful, such as implicitly defined lists in HTML documents, headings, and titles. One of ordinary skill in the art will recognize that semantically meaningful structures other than these structures may also be located by page analyzer 410.

OPERATION OF RANKING COMPONENT 225

[0045] Fig. 8 is a flow chart illustrating operations according to an exemplary implementation for determining ranking scores for documents. In one implementation, the ranking scores may be generated in the context of a search engine that is returning documents to a user that are relevant to a search query.

[0046] In other implementations, ranking component 225 may more generally be used to generate a ranking score for any document relative to a sequence of terms.

[0047] To begin, ranking component 225 may receive a document and a list of terms, such as a search query, that are to be used to rank the document (act 801). The document may be analyzed to locate semantically meaningful structures such as lists (implicit lists or explicit lists) or the text associated with headings/titles (act 802). As previously mentioned, this analysis may be performed by page analyzer component 410 and output as annotation information associated with the document. In some implementations, this analysis to locate semantically meaningful structures can be performed ahead of time on a corpus of documents. In this situation, act 802 may simply involve looking up the pre-calculated annotation information for the document.

[0048] Distance metrics may next be generated for pairs of terms in the search query (act 803). The distance metrics may be generated by, for example, distance component 420. The distance metrics may generally be based on how far apart terms are, such as that measured by the number of words ("word count") between terms, in the document. The distance metrics may be further

augmented by the concept of semantic closeness, determined based on structures within the document.

[0049] Fig. 9 illustrates concepts that may be used to augment the word count between terms when generating a distance metric. As shown, a list 900 includes a header 905 and three list items: item A 910, item B 912, and item C 914. Each item 910, 912, and 914 may include multiple words that describe the particular list item. Assume that both terms of a pair of terms are located in list 900. The following rules may be used to generate the distance metric between the terms: (1) if both terms appear in the same list item, the terms are considered close to one another; (2) if one term appears in a list item and the other term appears in header 905, this pair of terms may be considered to be approximately equally distant to another pair of terms that appear in header 905 and in another of the list items; (3) pairs of terms appearing in different list items may be considered to be farther apart than the pairs of terms falling under (1) and (2). As an example of the application of these rules, consider a pair of terms in which one term is the last word in item A 910 and the other term is the first word in item B 912.

Although these terms may be very close from a word count standpoint, the distance metric may indicate that they are farther apart than pairs of terms falling under (1) or (2).

[0050] For headings and titles, a term in the title of a document may be considered to be close to every other term in document regardless of the word count between the terms. Similarly, a term occurring in a heading may be considered to be very close to other terms that are below the heading in the tree structure.

[0051] Based on the distance metrics calculated for the pairs of terms in the search query, a ranking score may be calculated for the document (act 804). Relevance component 430 may, for example, calculate the ranking score for the document. In general, documents that include search terms that are relatively close to one another, as given by the distance metrics, may be given higher ranking scores (i.e., these documents are considered to be more relevant to the search query). In some applications, other factors could be used in generating the final ranking scores. For example, the inverse document frequency (IDF) may be used to give more weight to less common words. As another example, different measurements of document quality, such as those based on the interconnecting link-structure of a corpus of documents or those based on a predetermined quality of certain web sites or domains, may be factored into the final ranking score for a document.

EXEMPLARY IMPLEMENTATION

[0052] Fig. 10 is a diagram illustrating an exemplary implementation of ranking component 225 implemented in the context of an Internet search engine. A number of users 1005 may connect to a search engine 1010 over a network 1015, such as the Internet. Search engine 1010 may be a traditional search engine that returns a ranked set of documents related to a user query. Search engine 1010 may be a general search engine, such as one based on all documents from a large collection, such as documents on the web, or a more specialized search engine, such as a news search engine. In other implementations, search engine 1010 may be implemented over a specialized

corpus of documents, such as a corporate document database made available over a corporate network 1015.

[0053] In operation, search engine 1010 may receive a user query. The search engine may generate an initial set of documents that match the search query (i.e., documents that contain the terms of the search query) by comparing the search terms to document database 1020. The search engine may then submit the initial set of documents to ranking component 225 for processing. Ranking component 225, as described above, may generate ranking scores for the documents. Search engine 1010 may sort the set of documents based on the ranking scores. Search engine 1010 may then return some or all of the sorted documents (or references to the sorted documents) to the user.

CONCLUSION

[0054] Techniques for locating semantically meaningful structures in a document were described above. In one implementation, implicitly defined lists in an HTML document were located and used to augment a distance value defined for terms in the document. The augmented distance values may then be used in generating ranking scores for the documents.

[0055] It will be apparent to one of ordinary skill in the art that aspects of the invention, as described above, may be implemented in many different forms of software, firmware, and hardware in the implementations illustrated in the figures. The actual software code or specialized control hardware used to implement aspects consistent with the present invention is not limiting of the present invention. Thus, the operation and behavior of the aspects were described without reference to the specific software code -- it being understood that a

person of ordinary skill in the art would be able to design software and control hardware to implement the aspects based on the description herein.

[0056] The foregoing description of preferred embodiments of the present invention provides illustration and description, but is not intended to be exhaustive or to limit the invention to the precise form disclosed. Modifications and variations are possible in light of the above teachings or may be acquired from practice of the invention. For example, although many of the operations described above were described in a particular order, many of the operations are amenable to being performed simultaneously or in different orders to still achieve the same or equivalent results.

[0057] No element, act, or instruction used in the present application should be construed as critical or essential to the invention unless explicitly described as such. Also, as used herein, the article "a" is intended to potentially allow for one or more items. Further, the phrase "based on" is intended to mean "based, at least in part, on" unless explicitly stated otherwise. The scope of the invention is defined by the claims and their equivalents.